

Python! iPhone pics to fileGeoDB!

Contributed by Kevin Bell
 28, Apr. 2010
 Last Updated 28, Apr. 2010

The code below uses the Python Image Library to scrape the XYs from a geotagged iPhone pic and it creates a fgdb with a new folder within it named "geoPics". Your photos are copied to this folder and a hyperlink field is built in an output point feature class. You can uncomment the gp.getParameterAsText lines at the bottom to use in a Toolbox script tool, and when you set up the tool, all of the parameters are text and required.

This is tested on an iPhone, and other phones may have a slightly different way to store the XYs, so you may need to modify the _parseXYsFromExif function... The output projection is hardcoded.

```
#buildPointFCfromGeotaggedPhoto.py
#Author: Kevin Bell
#Date 4/2/2010
#Purpose:
#      Creates a file geodatabase from a directory of iPhone pictures
#      for best results, the phone should be on long enough to acquire
#      a gps fix.

#      The script will create a folder in your output fgdb named "geoPics" and
#      move your pics into it. You'll get a point feature class with a "hyperlink" field
#      that points to the picture in the new folder.

#      This script has a hardcoded output projection that you may want to change.

#      You need PIL, the Python Image Library. It has functions to scrape the XY's from
#      the exif header in your pic. The XY's come out a little differently for other cameras,
#      so if you're not using an iPhone you may need to modify the _parseXYsFromExif function.

## STILL NEED TO DO!!! :
##      -get the rotation data 17: (3873, 727)
##      -get the DOP data 11: (3, 1),
##      -get the timestamp data 7, but is there a date? 7: ((18, 1), (24, 1), (2646, 100))
```

```
import shutil
import os
import sys
import pprint
from PIL import Image
from PIL.ExifTags import TAGS
import arcgisscripting
gp = arcgisscripting.create(9.3)

def _get_exif(fn):
    """Extract all exif metadata"""
    ret = {}
    i = Image.open(fn)
    info = i._getexif()
    for tag, value in info.items():
        decoded = TAGS.get(tag, tag)
        ret[decoded] = value
    return ret

def _parseXYsFromExif(aDict):
    """parse out the GPS info from the exif data
    http://www.exiv2.org/tags.html"""
```

```

latDegree = aDict.get(2)[0][0]
latMin = aDict.get(2)[1][0]
latMin = latMin * 0.01

longDegree = aDict.get(4)[0][0]
longMin = aDict.get(4)[1][0]
longMin = longMin * 0.01

return latDegree, latMin, longDegree, longMin

def _parseNonXYexifData(aDict): ## Not implemented
    "get the rotation, DOP, and time here"
    ## this still isn't populating the fgdb
    try:
        hour = aDict.get(7)[0][0]
        minute = aDict.get(7)[1][0]
    except:
        pass

    try:
        DOPnum = aDict.get(11)[0][0]
        DOPdec = aDict.get(11)[0][1]
    except:
        pass

    ##rotation = the Numbers don't make sense
    ##return time, DOT
    pass

def _convertDegreeDecMin2DD(latD, latMin, longD, longMin):
    "convert From Degrees decimal-minutes (D m) to decimal-degrees (d) "
    #latD, latMin, longD, longMin = coordTuple

    latDD = latD + (latMin / 60)
    longDD = longD + (longMin / 60)

    return latDD, longDD

def _createFGDB(path, fgdbName, fcName):
    "create a fgdb and a fc with a field to contain the path to a picture"

    if gp.Exists(path + "\\" + fgdbName):
        print "...the file gdb already exists"
        pass
    else:
        gp.CreateFileGDB(path, fgdbName)
        print "...created the file gdb"

    if gp.Exists(path + "\\" + fgdbName + "\\" + fcName):
        print "...the fc already exists"
        pass
    else:
        spRef = r"Coordinate Systems\Geographic Coordinate Systems\World\WGS 1984.prj"
        gp.CreateFeatureClass_management (path + "\\" + fgdbName, fcName, "POINT",
                                         "#", "#", "#", spRef)
        print "...made fc"

        gp.AddField_management (path + "\\" + fgdbName + "\\" + fcName, "PicsPath", "TEXT",
                               "#", "#", "100", "#", "#", "#")
        print "...added field"

    os.mkdir(path + "\\" + fgdbName + "\\" + "geoPics")

```

```

print "...made geoPics dir in fgdb"

def makeGISpointsFromPics(inPicFolder, path, fgdb, fc):
    """Top level function... """

    print "----Beginning to build a GIS Point data for geotagged photos----"
    #build the geodatabase
    _createFGDB(path,fgdb, fc + "WGS84")
    dsc = gp.Describe(path + "\\" + fgdb + "\\" + fc + "WGS84")
    shpFld = dsc.ShapeFieldName

    pics = os.listdir(inPicFolder)
    pics = [p for p in pics if p.endswith(".JPG") or p.endswith(".jpg")]
    i = len(pics)
    for pic in pics:
        try:
            d = _get_exif(inPicFolder + "\\" + pic)
            print "\n"
            print pic
            pprint.pprint(d.get("GPSInfo"))

            latDegree, latMin, longDegree, longMin = _parseXYsFromExif(d.get("GPSInfo"))
            Lat, Lon = _convertDegreeDecMin2DD(latDegree, latMin, longDegree, longMin)

            ## theTime, theDOP = _parseNonXYExifData(d.get("GPSInfo"))

            pnt = gp.CreateObject("Point")
            pnt.Y = Lat
            pnt.X = Lon * -1

            rows = gp.InsertCursor(path + "\\" + fgdb + "\\" + fc + "WGS84")
            row = rows.NewRow()
            row.SetValue(shpFld, pnt)
            ##row.time
            ##row.DOP
            #row.PicsPath = "lkasjdflkjsdf"
            row.PicsPath = path + "\\" + fgdb + "\\geoPics" + "\\" + pic
            rows.InsertRow(row)

            # move from pic folder to fgdb\geoPics
            shutil.copy2(inPicFolder + "\\" + pic, path + "\\" + fgdb + "\\geoPics" + "\\" + pic)

            print "...added a point, " + str(i - 1) + " to go."
            i -= 1
        except:
            pass

    inCS = r"Coordinate Systems\Geographic Coordinate Systems\World\WGS 1984.prj"
    outCS = r"Coordinate Systems\Projected Coordinate Systems\State Plane\NAD 1983 (Feet)\NAD 1983 StatePlane Utah
Central FIPS 4302 (Feet).prj"
    gp.Project_management(path + "\\" + fgdb + "\\" + fc + "WGS84", path + "\\" + fgdb + "\\" + fc, outCS,
    "NAD_1983_To_WGS_1984_4", inCS)

    gp.Delete_management(path + "\\" + fgdb + "\\" + fc + "WGS84")

    print "-----done-----"

#-----
#inPicFolder = gp.getParameterAsText(0) # The path to your pics
#path = gp.getParameterAsText(1)      # The path to the output fgdb

```

```
#fgdb = gp.getParameterAsText(2)      # The name of your new fgdb
#fc = gp.getParameterAsText(3)        # The name of the point feature class of your pics

#makeGISpointsFromPics(inPicFolder, path, fgdb, fc)
makeGISpointsFromPics(r"N:\py\geotaggedPics\pics", r"D:\ztemp", "out9.gdb", "pics")
```